Method Association Approach: Situational construction and evaluation of an implementation method for software products

Rébecca Deneckère, Charlotte Hug, Juliette Onderstal, Sjaak Brinkkemper

Method Association Approach: Situational construction and evaluation of an implementation method for software products

Rébecca Deneckère[1], Charlotte Hug[1], Juliette Onderstal[2], Sjaak Brinkkemper[3]

[1]Centre de Recherche en Informatique
Université Paris 1 Panthéon-Sorbonne
Paris, France
rebecca.deneckere@univ-paris1.fr
charlotte.hug@univ-paris1.fr

[2]University Medical Center Utrecht
Utrecht, the Netherlands
j.onderstal@umcu.nl

[3]Department of Information and Computing Sciences
Utrecht University
the Netherlands
S.Brinkkemper@uu.nl

***Abstract*— Software implementation is one of the important steps in a software engineering process. It consists of integrating software based services or components in business alignment with the organizational view and acceptance from the users' perspectives. However, this step is complex and not supported in detail by the existing design and implementation methods. When implementing a software product in a customer organization with a specific context, the problem of the choice of the method or its adaptation is crucial to ensure the implementation success. Software producing organizations have difficulty with the creation of the most suitable implementation method for their software products. Situational Method Engineering (SME) proposes solutions to create methods adapted to the project at hand. We propose an approach to build an implementation method based on the association of method fragments, offering two advantages: it facilitates (a) the modeling of fragments by using the Process Deliverable Diagram formalism (PDD) that has proved its efficacy and simplicity, and (b) the selection of fragments by using metrics to analyze them. We illustrate our proposal with a case study to create an implementation method for a personal health management software product.**

Keywords*— *Software product implementation method, situational method engineering, method association, feature, method fragment, project situation

I. INTRODUCTION

Most software applications used by organizations in the public as well as in the private sector are standard software products developed by so-called software producing organizations (SPO) [1]. These software products (e.g. Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), Business Process Management Systems (BPMS), and Hospital Information Systems (HIS)) require to be implemented in a structured and careful manner by the SPO at the customer organizations of varying size, operations, and culture. However, SPOs are confronted with many obstacles in arranging a smooth implementation of their products into customer organizations [2][3]. Out of the many issues reported in literature, we mention alignment with business strategy [4] [5], integration with the infrastructure of other enterprise applications [6], organizational change and disruption [7], and diversity of maturity in ICT management [8].

The proposition of this paper is then to present a structured approach to develop methods for the implementation of software products. A software product implementation method is commonly defined as a "systematically structured approach to effectively integrate software based services or components into the workflow of an organizational structure or an individual end-user."[1].

According to [10], an implementation method "ensures a consistent and repeatable delivery of the software product, provide visibility to all the parties involved, to control costs and deliver to commitments made in terms of quality and time and a successful adoption of the software with the customers to ensure they derive the benefits of implementation and also provide a reference to other prospective customers in future."

Although many design and implementation methods exist, this set of possible methods decrease when focusing on very specific business domains (like health care, manufacturing, marketing, financial assets management). Moreover, even when several methods are available in the literature, most of the significant problems SPOs or customer organizations face occur during the implementation of the software products. In this work, we focus on the particularities of software implementations, which include the installation of the software and the training of the users at a customer organization. We also incorporated the evaluation, maintenance and support once the customer accepted the developed software.

The main research question is: how to create the most suitable implementation method for a specific software product?

[1] This definition, taken from http://en.wikipedia.org/wiki/Product_software_implementation_method, has been widely reused in research papers. It is worth noting that *implement* is defined as the fact "to start using a plan or system" [9], in this paper, *implementation* then means *deployment*.

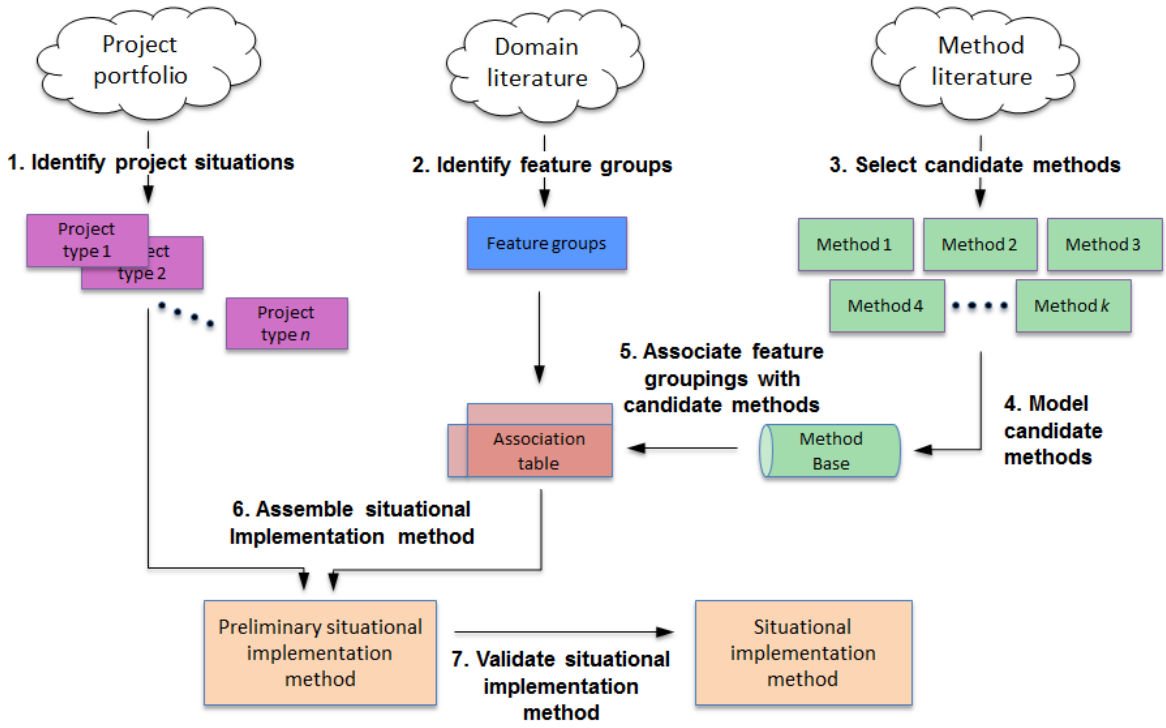

Fig. 1. Overview of the Method Association Approach.

Situational Method Engineering (SME), introduced in early nineties [11], argues that the method to be used for the development of an information or software system must be aligned with the context of the project because the engineering situation of each project is different and requires a different methodological support.

For this purpose, SME promotes the situation-specific method construction by reusing parts of existing methods generally designed as autonomous components and possibly stored in method repositories. Today, many different SME approaches exist (e.g. assembly-based [12] [13] [14], configuration-based [15], process tailoring [16], model driven engineering [17] and service-oriented [18]) but their implementation in practice appears to be difficult. Enterprises are slow to adopt the approaches and techniques proposed by SME researchers even though they acknowledge the significance of the role that methods play in their engineering activities. One of the main difficulties encountered by practitioners is the selection of the method components to create the situational method. As a matter of fact, the selection may be made with several criteria and the existing techniques are quite complicated.

We propose here a new approach (called Method Association Approach - MAA) to help engineers to create a suitable software implementation method adapted to their needs. This approach has been tested on the design of implementation methods for CMS based web applications [19] and is generalized here for any kind of implementation method. The generic process of the MAA is illustrated in Fig. 1.

The rest of the paper is organized as follows: section II presents the concepts of the Method Association Approach. In section III we introduce the formal matching analysis used to associate features group with method fragments (step 5 in Fig. 1), section IV describes the process of the MAA, Section V presents a case study with the evaluation of the obtained software implementation method using MAA and section VI concludes this paper.

II. Method Association Concepts

In this section we present the Method Association concepts in a metamodel (see Fig. 2). We divide it in four parts to ease its understanding and description. Note that the colors of the classes correspond to the colors of the output in Fig. 1.

A. *Project situation and characteristics*

Brinkkemper [20] and van de Weerd et al. [21] showed the importance to distinguish the situations of different projects, called "implementation situations" or "development project situations" [19]. Each customer software implementation corresponds to a specific project situation (several customer implementations may share the same characteristics, then the same project situation). These project situations are categorized by characteristics that are unique for a specific project. These characteristics are also called contingency factors or project factors [22]. Kornyshova et al. [25] proposes characteristics of IS development projects and categories as Organizational (Management commitment, Level of innovation…) or Development strategy (Delivery strategy, Project organization…) for instance. These characteristics allow identifying the project situation, for instance, a standard project situation will require little customization of the software whereas complex project situation will identify specific needs [19].

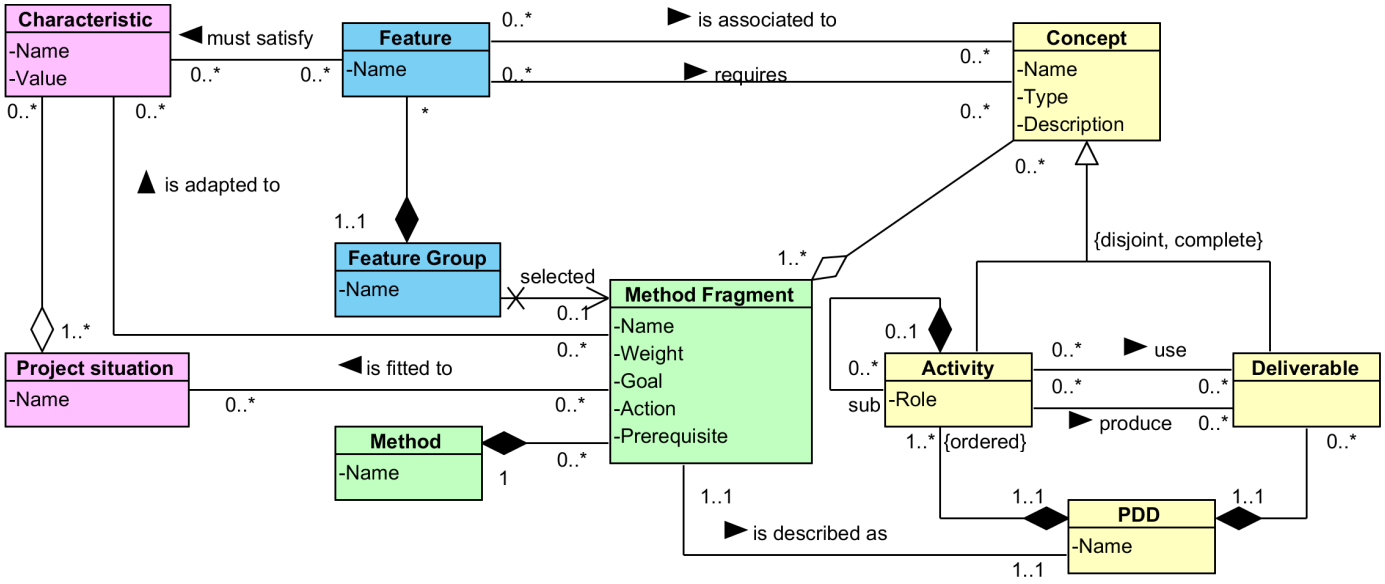

Fig. 2. Method Association concepts as a metamodel.

B. *Method and Method fragment*

For the selection, storage and assembling of the method fragments, a meta-modeling technique is necessary. According to Harmsen et al. [22], the technique should be able to include the formalization of "constraints or rules to avoid producing meaningless methods". A method base is used to capture knowledge of the candidate methods and to supply this knowledge [22]. The method fragments of existing methods in our approach are modeled as Process Deliverable Diagrams (PDD) [23]. A tool as the one developed in the incremental method engineering approach [24] could be used to automatically convert the PDDs into logic programs in disjunctive Datalog that generate optimal plans.

C. *Feature group and Subgroup*

Luinenburg et al. [19] defined feature groups as "a set of functional design requirements". These feature groups allow to compare the existing methods and selected fragments that meet the requirements of the product to be implemented. Features can be gathered from several sources: literature, field artifacts, expert interviews, etc. The number of features can be important but we can group them in specific sets called feature groups. A method fragment can be selected to meet the requirements of a particular feature group. Finally, features have to satisfy the characteristics that describe the constraints and situation of the current project.

A method fragment, according to our metamodel, must have a name, a weight, a goal, an action and may require prerequisites. We can complete the description of a method fragment with characteristics to describe the reuse situations.

D. *Activity, Deliverable, PDD and Concept*

A process-deliverable diagram is a meta-modelling technique developed for the method engineering process [21]. PDDs can be used for "analyzing, storing, selecting and assembling" method fragments [21]. It is represented as a diagram that describes activities and deliverables that act as input or output of these activities. A PDD is composed of two parts: the process model is on one side (activities) and the product model on the other (deliverables). We introduce the notion of Concept that can be an activity or a deliverable. An activity can be decomposed in sub-activities.

A Method Fragment, described as a PDD, is then composed of concepts. A feature can require a specific concept, for example, the "Planning" feature can require the activity "planning" or the deliverable "Gantt diagram". This requirement association will then help to associate features to method fragments. Moreover, a feature can be associated to a concept if it fulfills the requirements, to facilitate the selection of method fragments.

III. Formal Matching Analysis

The link between the features and the method fragments is found in different ways: (a) by identifying semantic associations between the terms used to describe them, (b) by studying the features requirements, (c) by studying the project situation and (d) by using the method engineer expertise.

A. *Semantic analysis*

As in the work of [14] using three similarity measures (synonymy, hyponymy and hyperonymy), we define several similarity metrics to be able to select the right components. First, we can compare the terms used in Feature, Feature group and method fragment. The name of the feature or feature group is compared to the name, goal or action of each fragment. Several possibilities can occur as the links between the terms can be of different nature. We defined metrics to identify these links. All metrics are defined as logical expressions described with a binary relation.

1) Equality

The metric EQ defines an association of equality between two terms Ti and Tj (Ti = Tj). If a feature and a fragment use the same terms, we identify it as an equality relation as follows: `EQ (Ti, Tj) = true` where Ti is a `Feature.name` or a `FeatureGroup.name`, Tj is a `MethodFragment.name`, a `MethodFragment.goal` or a `MethodFragment.action`.

For instance, a feature Data Conversion will be related to the fragment Data Conversion of the SDM method, as `EQ ("Data Conversion","Data Conversion")=true`.

2) Inclusion

The metric INC defines an inclusion association between two terms Ti and Tj (Ti ⊂ Tj) as follows: `INC (Ti, Tj)` where Ti is a `MethodFragment.name`, a `MethodFragment .goal` or a `MethodFragment.action` and Tj is a `Feature .name` or a `FeatureGroup.name`.

For instance, the feature Software Installation is related to the fragment Software of the MOOSAD method, as `INC ("Software","Software Installation")=true`.

3) Synonymy

The metric SYN defines a symmetric association between two terms Ti and Tj (Ti ≠ Tj) seen as synonyms (e.g. `SYN("user","stakeholder")=true)`.

4) Proximity

The metric PRO defines a semantic proximity association between two terms Ti and Tj (Ti ≠ Tj) (e.g. `PRO ("functional analysis","requirement")=true)`.

The first two metrics (Equality and Inclusion) can be easily used as they consist of the comparison of the written form of the terms. Synonymy and Proximity comprise a more important semantic analysis. This analysis may be automatized with the use of ontologies. We choose to use ConceptNet [26] that includes the term relationships of WordNet and Cyc; it then offers richer semantic links. This ontology includes definitions, lexical relationships and common sense associations that ordinary people make among concepts [27]. It defines all kind of association between terms, going from sheer equivalence to translation or composition. This proposal will use several of these associations to find links between the terms used in the features and those of the fragments. The metric SYN will use the *Synonym* and the *DefinedAs* relations of ConceptNet as <Engineer *DefinedAs* Application of Science>, <Process *Synonym* Task>. The metric PRO will use several kinds of ConceptNet relations as follows: (i) Hyponymy and Hypernymy defined by the IsA relation allow to manage the inheritance links between terms: <Task *IsA* Work activity>, and the *InstanceOf* relation allows to instantiate a concept <Microsoft *InstanceOf* Business> (ii) Meronymy and Holonymy: *PartOf* relation allows to handle the composition link between two terms as < goal *PartOf* Plan of action> (iii) Closeness: the *RelatedTo*, *DerivedFrom* and *Hasproperty* relations allow to find terms in close relation with another term as <Plan *RelatedTo* schedule>, <Requirement Analysis *DerivedFrom* Requirement>, <Test *HasProperty* Evaluation Method>.

More examples are shown in the case study section V.D.

B. Requirement feature analysis

As shown in Fig. 2, a feature may require concepts. For instance, the feature 'Planning' requires the concept of 'Schedule and Define tasks'. The method engineer can then define the links between the features and the method concepts by analyzing their nature. The link '*UsedFor'* of ConceptNet can be useful in this specific case to help the method engineer to identify some of these associations. The metric REQ defines a requirement link between a feature *F* and a method fragment *M* as follows: `REQ (F,M)=true`.

C. Project situation analysis

The situation itself can require the use of a specific method fragment. For instance, if the software implementation has to be customized for a specific customer organization, a design phase is then needed and the corresponding fragments have to be selected. The method engineer may then analyze the project situations and their characteristics to narrow the set of suitable fragments.

D. Heuristic analysis

Experts are needed to validate the links between fragments, feature groups and characteristics and find new ones. As a matter of fact, although the use of ontology allows to obtain many links between terms, they are not all meaningful for the project at hand. Experts are able to understand the links and to select a coherent sub-set. In a quite different way, ConceptNet does not contain all the possible links between terms and experts are needed to identify the missing ones. For instance, although the testing phase concept should be related to the concept of project management or project schedule, this term in ConceptNet is not related at all to these concepts but essentially to the terms of exam, school or examination. This link has to be identified by an expert of the domain to be taken into account into the formal matching analysis.

IV. Method Association Process

We specify here the proposed approach as a PDD (see Fig. 3), formally describing the overview given in Fig. 1. The method comprises six activities: Identify project situation (step 1 in Fig. 1), Feature grouping (step 2), Associate feature groups to concept (step 5), Select method fragment (step 6), Validate situational implementation method (step 7)and Create method fragment (steps 3 and 4)which is independent of the other activities.

A. Identify project situations

The first step is to identify the situations of the potential projects concerned by the software implementation. The stakeholders specify the characteristics of the project. Several project situations may exist for the same software implementation and each project will follow its own route within the method.

B. Features grouping

A domain expert defines the features required for the method to be built for the project under study. Luinenburg et al. [19] propose to use association criteria to be able to select the method fragments required for the situational method. These criteria are inspired by the Key Feature Groups of the software

product domain. To identify the feature groups, we propose to conduct interviews with experts, to study the literature and to analyze documents. The features are then categorized in groups following their occurrence in the obtained list.

C. Associate feature groups to concept

The features might require specific concepts (activity or deliverable). Then, the method engineer is able to associate the features to the most adequate concepts using the defined characteristics. There are two perspectives to associate feature groups to concepts. In their study, Luinenburg et al. [19] focused on the functionalities or deliverables that the method should contain because their goal was to develop a design method. This focus is called the "product perspective" [20]. The method fragments and features are then defined with a product perspective as deliverables. The "process perspective" [20] focus on the activities and on how the artifacts are produced. In their approach, Luinenburg et al. [19] introduced the association table for the comparison and the selection of the candidate method fragments (see example in TABLE III.). In this step, the feature groups are associated with the relevant method fragments of the method base. The rows of the association table represent the feature groups and their features and the column list the methods and their relevant concepts. Luinenburg et al. [19] presents two strategies for the qualitative analysis of the association results. The first one is the 'feature group strategy 'where the method fragments for the design or accomplishment of a feature group can be identified. The second strategy is the 'web modeling strategy': if a concept represents several feature groups this fragment might be relevant for the method under construction. On the contrary, when the concepts do not represent a feature group they might not be relevant for the method.

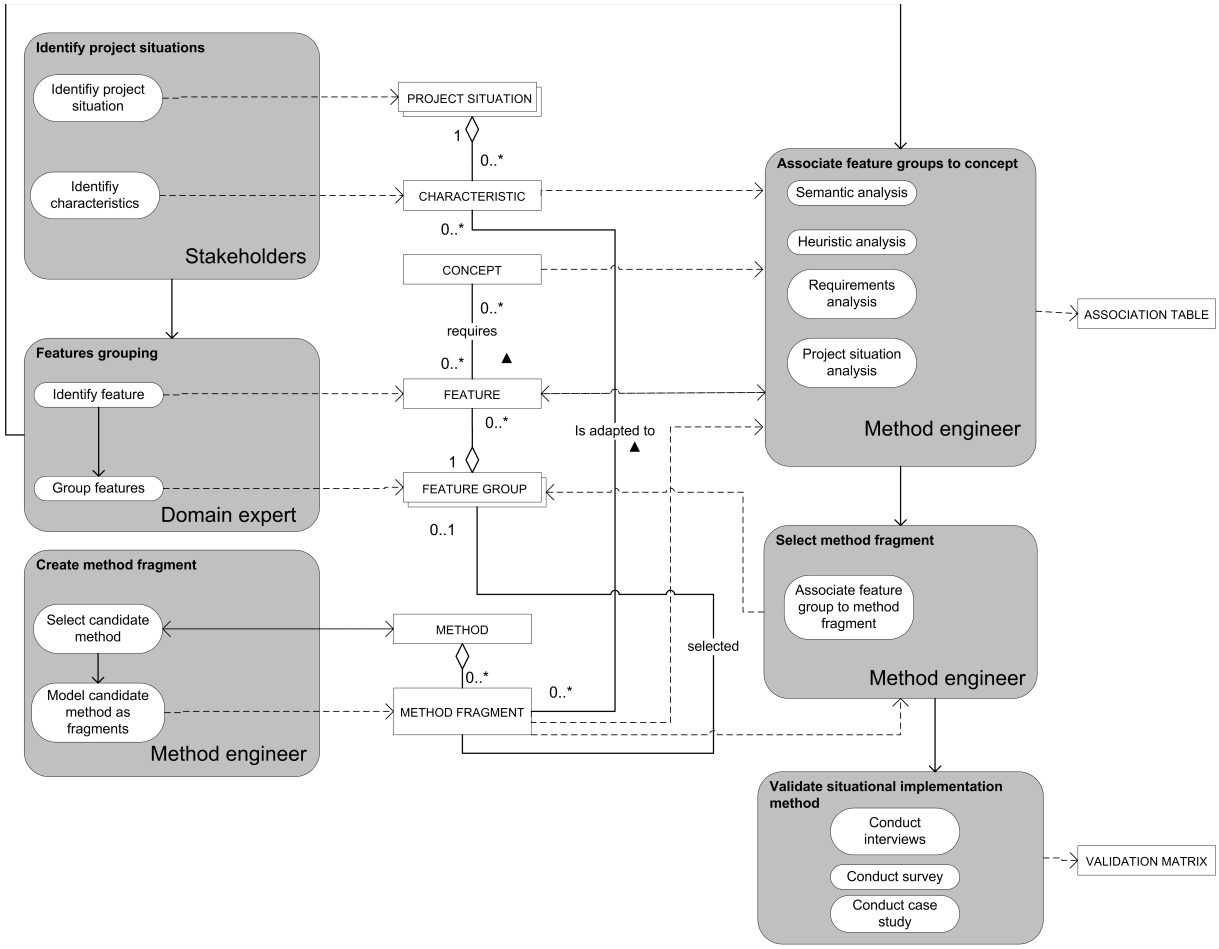

Fig. 3. PDD of the proposed approach.

D. Select method fragments

This step allows the method engineer to associate a feature group to the best-fitted method fragment, based on the associations between the features (or feature groups) and the concepts previously defined (the selected association between Feature or Feature group and Method fragment is then instantiated). The association table shows which fragments are most relevant, as they cover the feature groups. These fragments will be used to create the situational method.

E. Validate situational implementation method

The last step consists in evaluating the produced method based on surveys, interviews of experts or case studies. The internal quality of the developed method is evaluated through a matrix that gives an overview of the changes needed to

complete the method (see Table I). The left side of the matrix states the three possibilities of change: an existing activity (or sub-activity) might be removed or changed, or a new activity (or sub-activity) might be inserted. For each change, the matrix shows the impacted activities and indicates if the change is defined for the complete activity or part of it. The matrix also indicates if the change is permanent or only applicable in certain situations. The last row presents the motivation for the change and when it is not applicable.

TABLE I. VALIDATION MATRIX

Method change	Total	Removed	Changed	Inserted
Total changes				
Complete				
Partial				
Situational				
Permanent				
Motivation				

F. Create method fragment

This step is completely independent of the rest of the process, as it can be done well before the construction of the implementation method. The existing methods are gathered from literature and from the project domain (for example, we can use the methods already used in the organization). To narrow the number of candidate methods down, some selection criteria are necessary, which can be, for instance, the acceptation by the community, the level of tool support, etc. [19]. Modeling a candidate method consists in representing one or more fragments extracted from the method using the PDD formalism. Fragments must also comprise a name, weight, goal, action and may require prerequisites as describe in Fig. 2.

V. VITALHEALTH CASE STUDY

We used the proposed approach to develop and assemble a method for VitalHealth Software B.V, a Dutch company that develops and sells health care management (HCM) software. This HCM-organization was looking for an improved method for the delivery, testing and implementation of its software.

A. Identify project situations

Based on interviews and artifact study, we discovered three different project situations based on different characteristics (type of hosting, upgrading). We then divided the project of the HCM-organization in three kinds of implementations: Standard software implementation, Customized software implementation and Platform implementation. Each implementation will define its own route to follow within the method.

B. Features grouping

We identified the relevant features for the implementation of the HCM software product and categorized them in feature groups based on their appearance and relevance. We gathered the features from literature, field artifacts and expert interviews. The literature contains a lot of information about the features a health care or disease management system or application must have, but less about which features the implementation method should include. We choose three main works in the literature [28] [29] [30] but also used the documents and artifacts of the HCM-organization and the expert interviews to identify the features

A number of 126 features were identified for the implementation of health care software. We categorized these features in 18 groups and after informal interviews with two experts we combined several feature groups, resulting in 10 feature groups presented in Table II.

TABLE II. FEATURE GROUPS.

Features groups		
1	***Project management***	The planning of all the activities performed by the different project members is important to stay within the predefined time and budget. Reporting the process and the results to the management and to the customer should be included in the method.
2	***Infrastructure arrangements***	[7] indicated that, in the health care management domain, several infrastructure arrangements are important: the system needs a connection with internet and intranet, and the hosting arrangement should be taken care of.
3	***HCM software security***	Security is always an important part of a software implementation process but, in the health care domain, the security must be optimal for the patient's privacy. The access to patient data should be restricted to specific employees. The software itself should be secured as well by the use of certificates, passwords, etc.
4	***HCM software installation***	The installation of the software should be performed on a very precise and well planned way. In HC most of the institutes are open day and night so the installation should be fast. Additional software is often necessary and should be installed
5	***HCM system integration***	The integration of the health care software with other systems and software already installed is necessary; this is called the internal integration. The integration with systems outside the institutes is called the external integration [7]. [5] mentioned that for a good integration between systems a standardized terminology is important to prevent communication mistakes.
6	***Clinical data conversion***	When a new system or new software is implemented, old data should be taken up in the new system as well. Data conversion is necessary in this case.
7	***Health care professional and patient authorization***	Snyder et al. [8] indicated that the person identification is an important part of the system and that during the implementation of the software the right persons should get access to the right information. The patients might get access to their own data but not to all data and information.
8	***HCM system introduction***	To let the users within the health care organization work with the new software, proper training and documentation is important. Good communication with the customer and end users will ensure that the system will be properly used and that the users will better accept the new system.
9	***Project evaluation***	A good evaluation of the health care management software or system and the process of the project is "a must" in a complete implementation method.
10	***Support/ Maintenance***	To make sure the system will continue to work properly support has to be provided to the users and the system should be maintained. The health care software should be updated if necessary and all the versions that are modeled should be stored, if a new version is not working correctly

TABLE III. ASSOCIATION TABLE REPRESENTING THE FEATURE GROUPS AND THE CANDIDATE FRAGMENTS (EXCERPT)

		Unified Process							MOOSAD										SSA OnePoint							SDM										
		Planning	Getting Beta release out	Installing Beta release	Data migration or conversion	Complete artifacts	controlling transition process	Assess Transition phase	Testing	Developing System and User	Make Conversion Schedule	Hardware	Software	Conversion strategy	Data conversion	Revising management policies	Assessing Costs and Benefits	Motivating Adoption	Enabling Adoption	Deployment preparation	Production environment installation	Train trainers and end-users	Operational test run	Readiness review	Customer go-live	Phase closure review	Determine assumptions and Work Plan	Make Task description	Make Conversion- and implementation instructions	Give information and training	Convert data	Complement and document distribution	Make Exploitation- and production plan	Prepare work Environment and organization	Check preparation	Implementat new system
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
VitalHealth Case Study	1	Planning	X							X		X								X							X	X	X	X	X	X		X	X	
	2	Communication			X																									X		X	X	X	X	X	X
	3	Reporting			X				X														X			X	X				X			X	X	X
	4	Server installation										X								X	X						X							X		
	5	Client installation										X								X	X						X							X		
	6	Internet integration																		X	X												X			
	7	Intranet integration																		X	X												X			
	8	Hosting																																		
	9	Technology security																															X	X		
	10	Data security																														X	X	X		
	11	Software installation			X								X							X	X						X									X
	12	Additional software installation											X		X					X	X															
	13	Installation manuals		X	X	X				X																			X			X				X
	14	Software models					X																									X					X
	15	Internal				X									X					X	X											X				
	16	Shared care integration																															X			
	17	Conversion manner									X			X													X		X		X					
	18	Data conversion				X									X										X				X		X					
	19	User identification																			X															
	20	Patient identification																																		
	21	Roll specific																			X															

C. *Create method fragments*

As this step is independent from the others and as we needed method fragment to build the implementation software method for the case study, we decided to carry out this activity at this time of the study. The selected methods are the Unified Process [21], MOOSAD [22], SSA OnePoint Implementation Methodology (OPIM) - recently renamed Infor® Deployment Method , SDM [23] and GSDLC [24] as they include the notion of software implementation. We also included the already existing VitalHealth method in the HCM organization. We selected the methods in literature based on their level of detail, as we were looking for methods with clear and detailed steps and deliverables.

Another selection criterion was that the methods must describe the phases of the implementation process starting from the point that the software is delivered to the customer. Many methods found in literature only describe the analysis and development phases and not the next ones. The five selected methods meet these selection criteria. These methods were the only ones that we could find in the literature that describe the activities needed for the implementation of the software and not only its design.

All the main steps of the implementation of a system that are described in the methods were modeled in PDDs to get a quick overview of the activities and their related deliverables. These PDDs were stored in the method base to be used for the development of the situational method for the implementation of the software at the HCM organization. We finally defined 55 method fragments.

D. *Associate feature groups to concept*

In this project, we focused on the activities rather than on the deliverables because we wanted to develop a method for the implementation of the already designed and modeled software. The focus is not on the products and deliverables but on *how* the product is implemented at the customer. We are then in a "process perspective". After several interviews with consultants it became clear that the necessary improvements in the implementation process were not on what was produced but rather on how and when it was performed.

We then mapped the feature groups with the activity concept of the previously defined method fragments in an association table, using all the analysis metrics. Table III presents the previously defined method fragments, their concepts and the desired feature groups.

Crosses in Table III indicate the representation of the desired feature group in a method fragment. It means that there is an association link between the feature and the concept - Association(Feature, MethodFragment) = true. Below are shown some examples of metrics we used to build the association table. In these examples, Feature 1 is named F_1, Feature 2 is named F_2… Feature N is named F_N and MethodFragment 1 is named MF_1, Method Fragment 2 is named MF_2… MethodFragment N is named MF_N.

- F_1.name= "Planning" and MF_1.name= "Planning" then EQ (F_1.name, MF_1.name)=true and Association(F_1, MF_1)= true.
- F_{11}.name= "Software installation" and MF_{12}.name= "Software" then INC (MF_{12}.name, F_{11}.name)=true and Association(F_{11}, MF_{12})= true.
- F_{13}.name= "Installation manuals", MF_9.name= "Developing system and user documentation" and ConceptNet tells us that a "Manual" *IsA* "Document type", then we can infer a proximity link between these two terms: PRO (F_{13}.name, MF_9.name)=true so Association(F_{13}, MF_9)= true.
- F_{13}.name= "Installation manuals". The Feature 13 can be reached with a set of different process parts. There is a requirement feature link between this element and the method fragments MF_2, MF_3 and MF_4 (MF_2.name= "Getting beta release out", MF_3.name= "Installing beta release" and MF_4.name= "Data conversion and migration"). As REQ (F_{13}, MF_2)=true, REQ (F_{13},MF_3)=true and REQ (F_{13}, MF_4)=true then Association(F_{13}, MF_2)= true, Association(F_{13}, MF_3)= true and Association(F_{13}, MF_4)= true.
- F_2.name= "Communication", MF_{29}.name= "Give information and training". ConceptNet shows a *UsedFor* link between "Express information" and "Communication". This leads to the definition of the link REQ (F_2, MF_{29})=true and Association(F_2, MF_{29})= true.

The results showed that all the feature groups are associated to at least one method fragment, which means that the implementation method can be entirely created with the selected fragments.

For the qualitative analysis of the association results we used the "feature group strategy" [19]. We indicated for each required feature (or feature group) the corresponding activity of the method (when possible). For instance, the Unified Process and MOOSAD methods include an activity to produce installation manuals but they do not propose any activity for data security (no corresponding concept for the data security feature).

For the web modeling strategy, we indicated which activities of the candidate methods were relevant for the implementation method for the HCM-organization and which were not. For example, the Deployment preparation of the SSA OnePoint methodology is relevant for several features of the implementation method for Health care systems.

E. Select method fragment

The association table highlighted the correspondence between the fragments and the feature and feature groups. When several fragments could realize a feature group, we choose the fragment that included the higher number of features.

We used different tools to support the use of the developed software implementation method in the VitalHealth (VH-SIM) case study:

- MS Project was used for the planning of the implementation process from selling the software to the maintenance and support, presenting the phases and the main activities of the method, their duration, the start and end dates, and the deliverable.
- MS Excel was used to create a detailed planning of the implementation method. A project manager can easily create a detailed planning - with a start and end date per activity - can check this planning and change the status of their tasks.
- MS Word was used to make several templates for the deliverables of the created implementation method: the Project Proposal (which includes the project design and the financials); the Project Plan (which includes the scope of the project, the planning, the roles, the responsibilities and risk management); and the Implementation Plan (which includes an overview of all the steps that must be taken for the implementation comprising a training plan, a detailed installation plan, and the activities that the customer should perform before and during the implementation of the VitalHealth software)
- We also implemented a detailed planning in the VitalHealth Support System - a system developed by VitalHealth. It allows planning, hours accounting project administration and reports generation. The detailed planning comprises all the activities and sub-activities of the VH-SIM. It should be used in combination with the high-level planning in MS Project to keep the overview of the whole project.
- We described the VH-SIM in the VitalHealth Wiki as text, pictures and models. It can be used as background information when implementing the software of VitalHealth. With this wiki the employees within the organization can share their experiences and make comments when they found a change in the method.

F. Validate situational implementation method

We used three techniques to evaluate VH-SIM:

- Expert interviews to make sure all needed steps were added in the method;
- A practice-oriented case study to check whether the method was usable and whether all the steps necessary for the implementation were included in the method;
- A survey addressed to the employees of the HCM organization to present the method and validate the usability of the support and whether all steps were included.

Each technique is described further in details. To better evaluate the created software implementation method, we used a matrix presented in TABLE I.

1) Expert interviews

We interviewed five experts employees of VitalHealth: three are consultants for the implementation of the health care software and are responsible for the planning of the project and

the communication with the customer; one is a technical consultant responsible for the installation and configuration of the necessary servers and software; and the last one is a senior solution architect, having the same duties as the consultants but with more expertise of the technical aspects and the modeling of the software.

We showed the PDDs of the created method to the experts and asked them to review if all the necessary steps were present in the method, if the order of the steps and the named used in the method were right, if all the deliverables were present and what they think of the presentation of the method as PDDs.

We also asked them to give their opinion about the created method in MS Project and to look at the template of the Project Plan that we developed in MS Word.

We filled in the validation matrix after each interview. We then synthesized all the figures in one overview table.

A total of 37 changes were identified and processed in the new method (see Table IV). Three sub-activities were removed because they were not necessary (in the project situation) or were not correct. 22 sub-activities were changed (13 activities were misplaced in the process and names were adjusted for 9 activities). Experts indicated that 2 of the activities that were in the wrong place were performed according to the situation: one for an existing customer, the other for a new software version. 12 sub-activities were inserted (4 were inserted in specific situations: for custom made software, for a new software version or on customer request).

TABLE IV. RESULTS OF EXPERT INTERVIEWS.

Method change	Total	Removed	Changed	Inserted
Total changes	***37***	3	22	12
Complete	***9***	1	4	4
Partial	***28***	2	18	8
Situational	***4***	0	0	4
Permanent	***22***	3	22	8
Motivation		Not necessary	Changed order (13), changed name (9)	To complete the method

Overall, the experts were pleased with the method and indicated it was a good guideline for the implementation of the software of the HCM-organization. No major structural changes were necessary. We observed that the changes identified by the experts were scattered over the method and the changed activities were reduced. In total, 40% of the 92 activities and sub-activities were removed, changed or added.

The planning in MS Project was too much detailed, so most of the experts indicated that a planning on a higher level would be more usable. Apart from some small changes, the project plan was usable, clear and structured. A few experts indicated that they would like to use the method in MS Excel for the planning of a project.

2) Case study

In this section, we first present the case study and its results, then the threats to validity.

a) VH-SIM in the HCM-organization

The objective of the case study was to validate VH-SIM. For the case study, we followed a project in a large Dutch rehabilitation clinic[2] employing more than 550 people. This project included three parts:

- The first one was the integration of the care weight of a patient in the existing system that is the cost of care. The budgets of health care clinics in the Netherlands depend, since January 2009, on the amount of care needed by their patients. In other words, the 'healthcare weight' of the patient determines the budget of the clinic.
- The second part was an amelioration of an earlier project finished in 2007 for the treatment and monitoring of patients with a specific disease.
- The last part of this project was the replacement of the server.

All parts of the project are designed, modeled and performed by the HCM-organization. Because this project involves customized software based on the VitalHealth platforms, the route within the method is the custom software implementation route with conversion from the database included. First a testing/implementation planning was made by a consultant based on the VH-SIM. We evaluated this planning and interviewed the project members.

When the project was almost finished, we identified the changes that were made in VH-SIM to adapt it to the situation of the organization (see Table V). A total of 10 changes were made: 5 unnecessary sub-activities were removed, 5 activities were changed because they were performed by the customer (training and documentation development), 1 sub-activity was inserted ("install certificates").

TABLE V. METHOD CHANGES CASE STUDY.

Method change	Total	Removed	Changed	Inserted
Total changes	***10***	5	4	1
Complete	***3***	2	1	0
Partial	***7***	3	3	1
Situational	***9***	5	4	0
Permanent	***1***	0	0	1
Motivation		Not necessary, lack of time	Activity performed by the customer	To complete the method

The method was a good guideline for the planning of the Testing phase and the Implementation phase and most of the activities were performed according to the VH-SIM. But because the development of the software was an iterative process (the customer wanted more functionalities to be included) and sometimes the customer was late in the delivery of necessary inputs, it was hard to plan the implementation over time.

[2]For confidentiality reasons some information is not disclosed.

The most striking aspect in this case study was that the customer performed some training activities himself. The training planning was not entirely shared with VitalHealth. Because the customer wanted to perform parts of the implementation himself, most of the changes in the method were situational. Usually, in most projects, these activities are all performed by VitalHealth.

b) Threats to validity

Speer and Havasi [26] state that case studies investigators must maximize four aspects of validity: the construct validity, the internal validity, the external validity and the reliability. The internal validity doesn't apply here as the case study isn't an explanatory or causal study.

- Construct validity

To ensure the construct validity of the case study, we had to establish correct operational measures for the concepts being studied. We used input of several sources: interviews with the project members at VitalHealth at the end of the project, some informal talks with the project members during the project and the study of the documents that were delivered during the implementation.

- External validity

External validity is about the generalization of the method. In this work VH-SIM is situational, so it is specific for a very particular situation and for a certain domain. The external validity would be ensured if we generalize VH-SIM in the whole healthcare domain. In this case, more case studies should be performed for several projects in this domain.

- Reliability

The reliability is ensured if the operations of the case study can be performed again with other projects. We used a structured protocol that can be easily used again in other cases. The results of the case study are stored in the validation matrix and this matrix can be used in other case studies if needed.

3) Survey

To collect more data for the validation of the VH-SIM, we performed a paper survey with the employees of the HCM-organization after the presentation of the method and tools. 22 employees attended the presentation and we emailed the survey and the presentation to all the employees of the HCM-organization in the Netherlands (30 in total). We collected 15 filled-in surveys. Two surveys were deleted because they were not usable, as the respondent did not filled more than 4 answers. In total we used 13 surveys for the validation of VH-SIM, which represents 43 % of the employees. The distribution of these respondents over the department is a good representation for the HCM-organization: 4 consultants, 4 members of the modeling team, 4 members of the platform development team and 1 member of the management team.

The survey consisted of a list with the activities of the Delivery, Testing and Implementation phases. We asked each respondent to indicate whether the activity was correct or not and if that activity was always performed (permanent) or only in certain situations (situational), for instance in the specific situation of a project within a large organization. We also asked the respondents to give their opinion about the four tools used for the support of the method.

9 respondents did find the activities correct. 4 respondents indicated that 18 sub-activities were not correct, mostly because of their names. For the indication of the situation (partial or situational) of the sub-activities, the respondents' opinions were divided most of the time. They all agreed that the "Data conversion and entry" sub-activity was situational.

We asked the respondents to give their opinion about the tools proposed for the support of the VH-SIM. 14 employees responded to this part of the survey. 6 respondents were positive about the high level planning in MS Project, the rest did not have an opinion. The planning in MS Excel was usable and clear for 5 respondents but 2 respondents were very negative about it because it was confusing.

The respondents were divided about the VitalHealth Wiki: 5 were positive and 4 were negative. The negative reactions were mostly about the navigation through the different parts of the wiki. One respondent indicated that it was too easy to change the data and information within the wiki.

6 respondents were negative about the VitalHealth Support System, 3 were positive. The respondents motivated their opinion by saying that the support system is not user friendly; it includes too much detail and is confusing.

From the results of the survey we can conclude that (besides some discussion about the naming and terms used in the method) the activities and sub-activities were correct. One respondent indicated that in the case of the delivery of the platform to partners, the installation that will be performed by VitalHealth is not the most important aspect. For this route more research will be then needed.

Concerning the situation in which the activities are performed or not, the survey is not so clear. More research is necessary to indicate the situational factors within this method.

The respondents were divided about the support tools. The high level planning in MS Project is well received but the other tools need more attention before they can be used properly.

4) Internal validity of VH-SIM

To check whether the internal validity of VH-SIM was correct we used a static test introduced by [35]. In this test five issues are addressed: Completeness, Consistency, Efficiency, Reliability and Applicability.

a) Completeness

We checked the completeness of VH-SIM in the expert interviews, case study and survey. In the expert interviews 12 sub-activities were added, but overall the method was considered complete. The added sub-activities 'Make software set-up documentation' and 'Deliver software set-up documentation' are some examples. In the case study only one sub-activity was inserted, 'Install certificates'. According to the respondents of the survey no activities were missing. So overall we can conclude that the method after the insertion of the extra activities is complete.

b) Consistency

The name and terms used in VH-SIM have to be consistent and clear. After the expert interviews 8 activity names were changed, mostly to fit to the already used names at VitalHealth. Four respondents of the survey (all Platform team) also suggested some changes in names or terms, but overall the method is considered consistent and clear for the users.

c) Efficiency

The VH-SIM was only used in one case study. In this context, the method was efficient in the way that the planning was easier to specify. To check the efficiency (time and costs) the method should be performed in more projects. The projects in which the method will be used should then be compared with similar projects in the same field in which the method was not used.

d) Reliability

It is important that VH-SIM is semantically correct. According to the experts and the respondents of the survey, the activities and sub-activities are strongly interrelated and the activities are all meaningful.

e) Applicability

The applicability of a method means it has to be usable and feasible. In the case study, the VH-SIM was considered usable for the planning of a project. Unfortunately, lack of time made it sometimes difficult to perform all the activities. The experts indicated that the VH-SIM is usable in a project. For example, one expert indicated that the use of this method as a checklist was useful, as no important steps (like the installation of the certificates, etc.) could be forgotten.

VI. CONCLUSION

We proposed in this work a new SME approach to create implementation methods for software products, named the Method Association Approach. We detailed the concepts and the steps of the approach theoretically. MAA uses analysis metrics to select the fragments matching the situation at hand. We illustrated MAA enactment on a real case, VitalHealth Software providing an evaluation of the produced software implementation method, VH-SIM.

The evaluation of VH-SIM gave positive results, so the Method Association Approach used to build VH-SIM is indirectly validated as its product (VH-SIM) has been validated. However, we must conduct a validation of each step of MAA to evaluate its process part. This validation requires the participation of method engineers.

One of the already identified difficult tasks is the identification of the feature and feature groups as it is quite long and complex. One of our perspectives is to work further on the ConceptNet ontology in order to develop the relations needed by the domain of software implementation method. This specific ontology will greatly improve the semantic analysis and facilitate the work of the experts and method engineers.

REFERENCES

[1] S. Jansen, S. Brinkkemper and J. Souer, Lutzen Luinenburg. "Shades of gray: Opening up a software producing organization with the open software enterprise model". Journal of Systems and Software 85(7): 1495-1510, 2012

[2] D. Robey, J.W. Ross and M.C. Boudreau, "Learning to implement enterprise systems: An exploratory study of the dialectics of change". Journal of Management Information Systems, 19(1), 2002, pp. 17-46

[3] F. Fui-Hoon Nah, J. Lee-Shang Lau and J. Kuang, "Critical factors for successful implementation of enterprise systems". Business process management journal, 7(3), 2001, pp. 285-296

[4] M.L. Markus, C. Tanis and P.C. Van Fenema, "Enterprise resource planning: multisite ERP implementations". Communications of the ACM, 43(4), 2000, pp 42-46

[5] D. Maditinos, D. Chatzoudes and C. Tsairidis, "Factors affecting ERP system implementation effectiveness". Journal of Enterprise Information Management 25(1), 60-78, 2011

[6] T.H. Nguyen, J.S. Sherif and M. Newby "Strategies for successful CRM implementation". Information Management & Computer Security, 15(2), 2007, pp. 102-115

[7] D. J. Finnegan and W. L. Currie, "A multi-layered approach to CRM implementation: An integration perspective", European Management Journal, Volume 28, Issue 2, 2010, pp. 153-167,

[8] M. Rohloff, "Advances in business process management implementation based on a maturity assessment and best practice exchange".Information Systems and e-Business Management, 9(3), 2011, pp. 383-403

[9] http://dictionary.cambridge.org, consulted in November 2014.

[10] M. Varadaraj, and N. Goud, "Successful Software Adoption - A study of Software Implementation Methodologies". International Journal of Computer Applications, vol. 41(16), 2012, pp. 42-47.

[11] K. Kumar, and R.J. Welke, "Methodology EngineeringR: A Proposal for Situation Specific Methodology Construction". In: Challenges and Strategies for Research in Systems Development, Cotterman, W. and J. Senn (eds.), J. Wiley, Chichester, UK, 1992, pp. 257-266.

[12] S. Brinkkemper, M. Saeki, and F. Harmsen, "Assembly Techniques for Method Engineering". In: Proceedings of CAiSE 1998, LNCS 1413, Springer Verlag 1998, pp. 381-400.

[13] Firesmith, D.G., Henderson-Sellers, B.: The OPEN Process Framework: An Introduction. Addison-Wesley, London, UK, 330pp 2002.

[14] J. Ralyté, and C. Rolland, "An Assembly Process Model for Method Engineering". In: Proceedings of CAISE 2001, LNCS 2068, Springer, Berlin, 2001, pp. 267-283.

[15] F. Karlsson, and P.J. Ågerfalk , "Method Configuration: Adapting to Situational Characteristics While Creating Reusable Assets". Inf. and Soft. Technology, vol. 46(9), 2004, pp. 619–633.

[16] M. Rossi, B. Ramesh, K. Lyytinen, and J.-P. Tolvanen, "Managing evolutionary method engineering by method rationale". Journal of the AIS, vol. 5(9), 2004, pp. 356-391.

[17] M. Cervera, M. Albert , V. Torres , and V. Pelechano, "A Model-Driven Approach for the Design and Implementation of Software Development Methods". International Journal of Information System Modeling and Design (IJISMD), vol. 3(4), 2012, pp. 86-103.

[18] G. Guzélian, and C. Cauvet, "SO2M: Towards a Service-Oriented Approach for Method Engineering". In: Proceedings of IKE'07, Las Vegas, Nevada, USA 2007.

[19] L. Luinenburg, S. Jansen, J. Souer, S. Brinkkemper, and I. van de Weerd, I., "An Approach to Creating Design Methods for the Implementation of Product Software: The Case of Web Information Systems". Advances in Conceptual Modeling – Challenges and Opportunities, Lecture Notes in Computer Science, vol. 5232, 2008, pp 426-436.

[20] S. Brinkkemper, "Method engineering: engineering of information systems development methods and tools, Information and Software Technology", vol. 38, 1996, pp. 275-280.

[21] I. van de Weerd, S. Brinkkemper, J. Souer and, J.A. Versendaal, "Situational Implementation Method for Web-based Content Management System-applications: Method Engineering and Validation in Practice". Software Process Improvement and Practice, vol. 11, 2006, pp. 521-538.

[22] A.F. Harmsen, S. Brinkkemper, and J.L.H Oei, “Situational method engineering for infor-mation systems project approaches”. In proceedings of the international IFIP WG8. 1 Con-ference in CRIS series: "Methods and associated Tools for the Information Systems Life Cycle" (A-55), North Holland 1994.

[23] I. van de Weerd and S. Brinkkemper, “Meta-modeling for situational analysis and design methods”. In M.R. Syed and S.N. Syed (Eds.), Handbook of Research on Modern Systems Analysis and Design Technologies and Applications, Hershey: Idea Group Publishing, 2008, pp. 38-58

[24] K. Vlaanderen, F. Dalpiaz and S. Brinkkemper,” Finding Optimal Plans for Incremental Method Engineering”. CAiSE 2014, pp. 640-655

[25] E. Kornyshova, R. Deneckère, and C. Salinesi, “Method Chunks Selection by Multicriteria Techniques: an Extension of the Assembly-based Approach”. Situational Method Engineering, 2007, pp. 64-78.

[26] R. Speer, and C. Havasi, “ Representing General Relational Knowledge in ConceptNet 5”. LREC, 2012, pp. 3679–3686.

[27] C. Junpeng,and L. .Juan, “Combining ConceptNet and WordNet for Word Sense Disambiguation”, Proceedings of the 5th International Joint Conference on Natural Language Pro-cessing, Chiang Mai, Thailand 2011.

[28] I. Iakovidis, “Towards personal health record: current situation, obstacles and trends in implementation of electronic health care record in Europe”, International Journal of Medical Informatics, vol.52, 1998, pp.105–115.

[29] W. Raghupathi, and J. Tan, “Strategic IT Applications in Health Care”. Communication of the ACM, vol. 45(12), 2002, pp. 56-61.

[30] K.D. Snyder, and P. Paulson, “Health care Information Systems: Analysis of Health care Software”. Hospital topics: Research a Perspectives on Health care, vol. 80 (4), 2002, pp. 5-12.

[31] I. Jacobson, G. Booch, and J. Rumbaugh, The Unified Software Development Process. Mas-sachusetts: Addison-Wesley 1999.

[32] A. Dennis, B. Wixom, and, E. Tegarden, Systems Analysis and Design with UML Verson 2.0 An Object-Oriented Approach. Hoboken, NJ: Wiley 2005.

[33] W.S. Turner, R.P. Langerhorst, G.F Hice, H.B. Eilers and, A.A. Uijttenbroek, SDM, System Development Methodology, Rijswijk: Cap Gemini Publishing 1990.

[34] P.O. Flaatten, D.J. McCubbrey, and P.D. O’Riordan, and K. Burgess, Foundations of Business Systems, Orlando: The Dryden Press 1992.

[35] R.K. Yin, Case Study Research- Design and Methods. SAGE Publications, Inc, London, United Kingdom, 2009.

[36] S. Brinkkemper, M.Saeki, and F. Harmsen, “Meta-Modelling Based Assembly Techniques for Situational Method Engineering”. Information Systems, vol. 24(3), 1999, pp 209-228.